**Area, Centroid and Area Moments for Polygonal Objects**

Summations shown below are for closed CCW boundary sequences. CW boundary sequences will produce negative values for area and moments. Closed boundaries require .

The term ai is twice the signed area of the elementary triangle formed by (xi,yi) and (xi+1,yi+1) and the origin.

For improved accuracy, a temporary local origin at the mean of the boundary points should be used.

















centroidal moments



principal moments





**Object with Holes**

1) Digitize the outline of the object in the CCW direction. Be certain to close the outline (i.e. the first and last points must be the same).

2) Digitize outlines of holes in the CW direction. Be certain to close the outlines (i.e. the first and last points must be the same).

3) Append the data strings. Remember to addend the first point in the outline to the end of each string.

4) Repeat steps 2) and 3) for multiple holes. For example, an object with three holes will contain a closed CCW outline for the object, followed by three closed CW outlines, one for each hole. Outlines for holes must be separated by the first point for the outline.

5) Area, centroid and moment computations will be correct. Perimeter will NOT be correct.

**X**

**Y**

**Sample data for figure at right**

x\_outline = [ 0 3 3 0 0 ];

y\_outline = [ 0 0 3 3 0 ];

x\_hole = [ 1 1 2 2 1 ];

y\_hole = [ 1 2 2 1 1 ];

x = [ x\_outline x\_hole x\_outline(1) ];

y = [ y\_outline y\_hole y\_outline(1) ];

**Sample data for three holes**

x = [ x\_outline x\_holeA x\_outline(1) x\_holeB x\_outline(1) x\_holeC x\_outline(1) ];

y = [ y\_outline y\_holeA y\_outline(1) y\_holeB y\_outline(1) y\_holeC y\_outline(1) ];

% test\_polygeom.m - test polygeom

% area, centroid, perimeter and area moments of polygonal outline

% H.J. Sommer III – 16.12.09 - tested under MATLAB v9.0

clear

% constants

d2r = pi / 180;

% 3x5 test rectangle with long axis at 30 degrees

% area=15, x\_cen=3.415, y\_cen=6.549, perimeter=16

% I1=11.249, I2=31.247, J=42.496

x = [ 2.000 0.500 4.830 6.330 ]';

y = [ 4.000 6.598 9.098 6.500 ]';

% get geometry

[ geom, iner, cpmo ] = polygeom( x, y );

% show results

area = geom(1);

x\_cen = geom(2);

y\_cen = geom(3);

perimeter = geom(4);

disp( [ ' ' ] )

disp( [ '3x5 test rectangle with long axis at 30 degrees' ] )

disp( [ ' ' ] )

disp( [ ' area x\_cen y\_cen perim' ] )

disp( [ area x\_cen y\_cen perimeter ] )

I1 = cpmo(1);

angle1 = cpmo(2);

I2 = cpmo(3);

angle2 = cpmo(4);

disp( [ ' ' ] )

disp( [ ' I1 I2' ] )

disp( [ I1 I2 ] )

disp( [ ' angle1 angle2' ] )

disp( [ angle1/d2r angle2/d2r ] )

% plot outline

xplot = x( [ 1:end 1] );

yplot = y( [ 1:end 1] );

rad = 10;

x1 = [ x\_cen-rad\*cos(angle1) x\_cen+rad\*cos(angle1) ];

y1 = [ y\_cen-rad\*sin(angle1) y\_cen+rad\*sin(angle1) ];

x2 = [ x\_cen-rad\*cos(angle2) x\_cen+rad\*cos(angle2) ];

y2 = [ y\_cen-rad\*sin(angle2) y\_cen+rad\*sin(angle2) ];

plot( xplot,yplot,'b', x\_cen,y\_cen,'ro', ...

x1,y1,'g:', x2,y2,'g:' )

axis( [ 0 rad 0 rad ] )

axis square

% bottom of test\_polygeom.m

function [ geom, iner, cpmo ] = polygeom( x, y )

%POLYGEOM Geometry of a planar polygon

%

% POLYGEOM( X, Y ) returns area, X centroid,

% Y centroid and perimeter for the planar polygon

% specified by vertices in vectors X and Y.

%

% [ GEOM, INER, CPMO ] = POLYGEOM( X, Y ) returns

% area, centroid, perimeter and area moments of

% inertia for the polygon.

% GEOM = [ area X\_cen Y\_cen perimeter ]

% INER = [ Ixx Iyy Ixy Iuu Ivv Iuv ]

% u,v are centroidal axes parallel to x,y axes.

% CPMO = [ I1 ang1 I2 ang2 J ]

% I1,I2 are centroidal principal moments about axes

% at angles ang1,ang2.

% ang1 and ang2 are in radians.

% J is centroidal polar moment. J = I1 + I2 = Iuu + Ivv

% H.J. Sommer III - 16.12.09 - tested under MATLAB v9.0

%

% sample data

% x = [ 2.000 0.500 4.830 6.330 ]';

% y = [ 4.000 6.598 9.098 6.500 ]';

% 3x5 test rectangle with long axis at 30 degrees

% area=15, x\_cen=3.415, y\_cen=6.549, perimeter=16

% Ixx=659.561, Iyy=201.173, Ixy=344.117

% Iuu=16.249, Ivv=26.247, Iuv=8.660

% I1=11.249, ang1=30deg, I2=31.247, ang2=120deg, J=42.496

%

% H.J. Sommer III, Ph.D., Professor of Mechanical Engineering, 337 Leonhard Bldg

% The Pennsylvania State University, University Park, PA 16802

% (814)863-8997 FAX (814)865-9693 hjs1-at-psu.edu www.mne.psu.edu/sommer/

% begin function POLYGEOM

% check if inputs are same size

if ~isequal( size(x), size(y) ),

error( 'X and Y must be the same size');

end

% temporarily shift data to mean of vertices for improved accuracy

xm = mean(x);

ym = mean(y);

x = x - xm;

y = y - ym;

% summations for CCW boundary

xp = x( [2:end 1] );

yp = y( [2:end 1] );

a = x.\*yp - xp.\*y;

A = sum( a ) /2;

xc = sum( (x+xp).\*a ) /6/A;

yc = sum( (y+yp).\*a ) /6/A;

Ixx = sum( (y.\*y +y.\*yp + yp.\*yp).\*a ) /12;

Iyy = sum( (x.\*x +x.\*xp + xp.\*xp).\*a ) /12;

Ixy = sum( (x.\*yp +2\*x.\*y +2\*xp.\*yp + xp.\*y).\*a ) /24;

dx = xp - x;

dy = yp - y;

P = sum( sqrt( dx.\*dx +dy.\*dy ) );

% check for CCW versus CW boundary

if A < 0,

A = -A;

Ixx = -Ixx;

Iyy = -Iyy;

Ixy = -Ixy;

end

% centroidal moments

Iuu = Ixx - A\*yc\*yc;

Ivv = Iyy - A\*xc\*xc;

Iuv = Ixy - A\*xc\*yc;

J = Iuu + Ivv;

% replace mean of vertices

x\_cen = xc + xm;

y\_cen = yc + ym;

Ixx = Iuu + A\*y\_cen\*y\_cen;

Iyy = Ivv + A\*x\_cen\*x\_cen;

Ixy = Iuv + A\*x\_cen\*y\_cen;

% principal moments and orientation

I = [ Iuu -Iuv ;

-Iuv Ivv ];

[ eig\_vec, eig\_val ] = eig(I);

I1 = eig\_val(1,1);

I2 = eig\_val(2,2);

ang1 = atan2( eig\_vec(2,1), eig\_vec(1,1) );

ang2 = atan2( eig\_vec(2,2), eig\_vec(1,2) );

% return values

geom = [ A x\_cen y\_cen P ];

iner = [ Ixx Iyy Ixy Iuu Ivv Iuv ];

cpmo = [ I1 ang1 I2 ang2 J ];

% bottom of polygeom